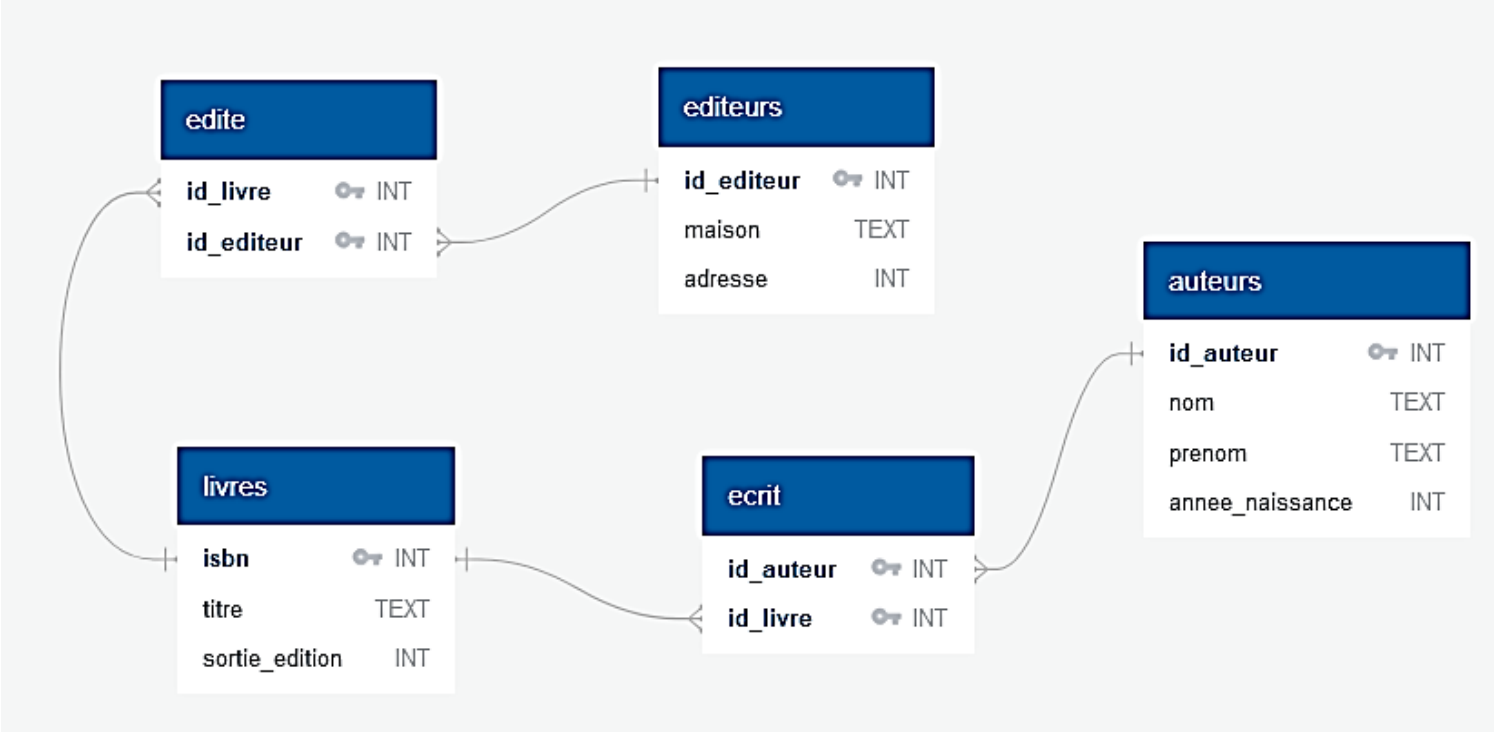
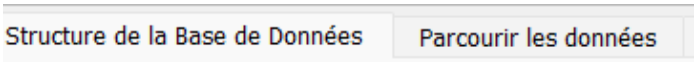
Tp _3 Gestion d'une bibliothèque

L'objectif de ce travail est d'exploiter la bdd existante d'une bibliothèque, bdd dont le diagramme relationnel est présenté ci-dessous :



Cette bdd est contenue dans le fichier *bibliotheque.db* à télécharger sur *nsibranly.fr* et à placer dans votre répertoire de travail.

⇨ Lancer le logiciel *DB Browser* et **ouvrir la base de données** contenue dans le fichier *bibliotheque.db*

Les onglets 

vous donnent alors respectivement la structure des 5 tables qui composent cette bdd et le contenu des celle-ci.

Les commandes Sql qui ont permis de définir ces tables sont données ci-dessous, mais aussi sur la page suivante avec une taille plus appropriée à leur lecture. On y retrouve les contraintes de clés primaires et étrangères :

```
CREATE TABLE "auteurs" ( "id_auteur" INTEGER PRIMARY KEY AUTOINCREMENT, "nom" TEXT, "prenom" TEXT, "annee_naissance" INTEGER )
CREATE TABLE "ecrit" ( "id_auteur" INTEGER, "id_livre" INTEGER, FOREIGN KEY("id_livre") REFERENCES "livres"("isbn"), FOREIGN KEY("id_auteur") REFERENCES "auteurs"("id_auteur"), PRIMARY KEY("id_auteur","id_livre") )
CREATE TABLE "edite" ( "id_livre" INTEGER, "id_editeur" INTEGER, FOREIGN KEY("id_livre") REFERENCES "livres"("isbn"), PRIMARY KEY("id_livre","id_editeur"), FOREIGN KEY("id_editeur") REFERENCES "editeurs"("id_editeur")
CREATE TABLE "editeurs" ( "id_editeur" INTEGER PRIMARY KEY AUTOINCREMENT, "maison" TEXT, "adresse" TEXT )
CREATE TABLE "livres" ( "isbn" INTEGER PRIMARY KEY AUTOINCREMENT, "titre" TEXT, "sortie_edition" INTEGER )
```

Les contenus des 5 tables sont donnés sur la page qui suit.

Les requêtes demandées dans la suite sont à écrire dans un fichier nommé *bibliotheque.sql* en utilisant Visual Studio Code. Le format d'écriture sera le suivant :

```sql
-- Requete 1 :
SELECT nom,prenom, MAX (annee_naissance)
FROM auteurs;

-- Requete 2 :
```

: "auteurs" ("id_auteur" INTEGER PRIMARY KEY AUTOINCREMENT, "nom" TEXT, "prenom" TEXT, "annee_naissance" INTEGER)
: "ecrit" ("id_auteur" INTEGER, "id_livre" INTEGER, FOREIGN KEY("id_livre") REFERENCES "livres"("isbn"), FOREIGN KEY("id_auteur") REFERENCES "auteurs"("id_auteur"), PRIMARY KEY("id_auteur","id_livre"))
: "edite" ("id_livre" INTEGER, "id_editeur" INTEGER, FOREIGN KEY("id_livre") REFERENCES "livres"("isbn"), PRIMARY KEY("id_livre","id_editeur"), FOREIGN KEY("id_editeur") REFERENCES "editeurs"("id_editeur"))
: "editeurs" ("id_editeur" INTEGER PRIMARY KEY AUTOINCREMENT, "maison" TEXT, "adresse" TEXT)
: "livres" ("isbn" INTEGER PRIMARY KEY AUTOINCREMENT, "titre" TEXT, "sortie_edition" INTEGER)

edite

id_livre	id_editeur
Filtre	Filtre
4	4
6	4
5	4
2	3
2	2
3	6
7	5
10	2
1	1
8	2
9	2
9	7
11	9
12	8

editeurs

id_editeur	maison	adresse
Filtre	Filtre	Filtre
1	Fluide Glacial	33 avenue du Maine, Paris 15
2	Livre de poche	43 rue de Grenelle, Paris 15
3	Fallois	22 rue de la Boétie, Paris 8...
4	Presse Pocket	12 avenue de l'Italie, Paris 13
5	Lombard	15-27 rue Moussorgski, Paris 18
6	Arche	15-27 rue Moussorgski, Paris 18
7	Casterman	66 rue Bonaparte, Paris 6
8	Anne Carrière	66 rue Bonaparte, Paris 6
9	IRMA	3333 rue Trifouille, Les Oies

auteurs

id_auteur	nom	prenom	annee_naissance
Filtre	Filtre	Filtre	Filtre
1	Franquin	André	1925
2	Eddings	David	1931
3	Volkoff	Vladimir	1932
4	Bond	Edward	1934
5	Hugo	Victor	1802
6	Brume	Annie	1971

livres

isbn	titre	sortie_edition
Filtre	Filtre	Filtre
1	Idées noires	1931
2	Le grand star blanc	NULL
3	Lear/ la mer	1953
4	Le pion blanc des présages	1895
5	Métro pour l'enfer	1964
6	La reine des sortilèges	1845
7	Alexandre Nevski	1820
8	L'homme qui rit	2001
9	Les travailleurs de la mère	1842
10	Hernani	1830
11	Le réseau	1974
12	Le réseau	1984

ecrit

id_auteur	id_livre
Filtre	Filtre
1	1
3	2
4	3
2	4
3	5
2	6
3	7
5	8
5	10
2	11
6	12
5	9

1- QUELQUES REQUETES SIMPLES :

Donner les requêtes Sql qui permettent :

Requête 1. : de retourner le nom et le prénom de l'auteur le plus jeune

```
SELECT nom, prenom,  MAX (annee_naissance)
FROM auteurs;
```

Résultat attendu dans DB Browser :

	nom	prenom	MAX (annee_naissance)
1	Brume	Annie	1971

Requête 2. : de retourner le nombre d'auteurs nés au 20E siècle (utiliser un alias pour avoir le retour ci-dessous)

```
SELECT COUNT(*) AS AUTEURS_20ème
FROM auteurs
WHERE annee_naissance >= 1900;
```

Résultat attendu dans DB Browser :

	AUTEURS_20ème
1	5

Requête 3. : de retourner le nombre de maison d'édition qui sont localisée à Paris (utiliser un alias pour avoir le retour ci-dessous)

```
SELECT COUNT(*) AS Parisienne
FROM editeurs
WHde mERE adresse LIKE "%Paris%";
```

Résultat attendu dans DB Browser :

	Parisienne
1	8

Requête 4. : de mettre à jour la date de naissance de Brume Annie : 1967 au lieu de 1971 indiqué

```
UPDATE auteurs
SET
annee_naissance = 1967
WHERE nom ="Brume";
```

2- INSERTION D'UN NOUVEL OUVRAGE DANS LA BDD :

François Villon né en 1431 et mort après 1463, est un poète français de la fin du Moyen Âge.

Dans les décennies qui suivent la disparition de Villon, son œuvre est publiée et connaît un grand succès.

Œuvres principales : « Le Lais » et « Le testament ».

Livres réédités par « Livre de poche » 43 rue de Grenelle, Paris 15, en 1972.

a. ECRITURE « A LA MAIN » DES NOUVEAUX ENREGISTREMENTS :

L'ajout de ces 2 nouveaux ouvrages se traduit par de nouveaux enregistrements à insérer dans les différentes tables de la bdd.

⇨ Rajouter au crayon ces enregistrements, en complétant les tableaux des contenus de la bdd, page 2.

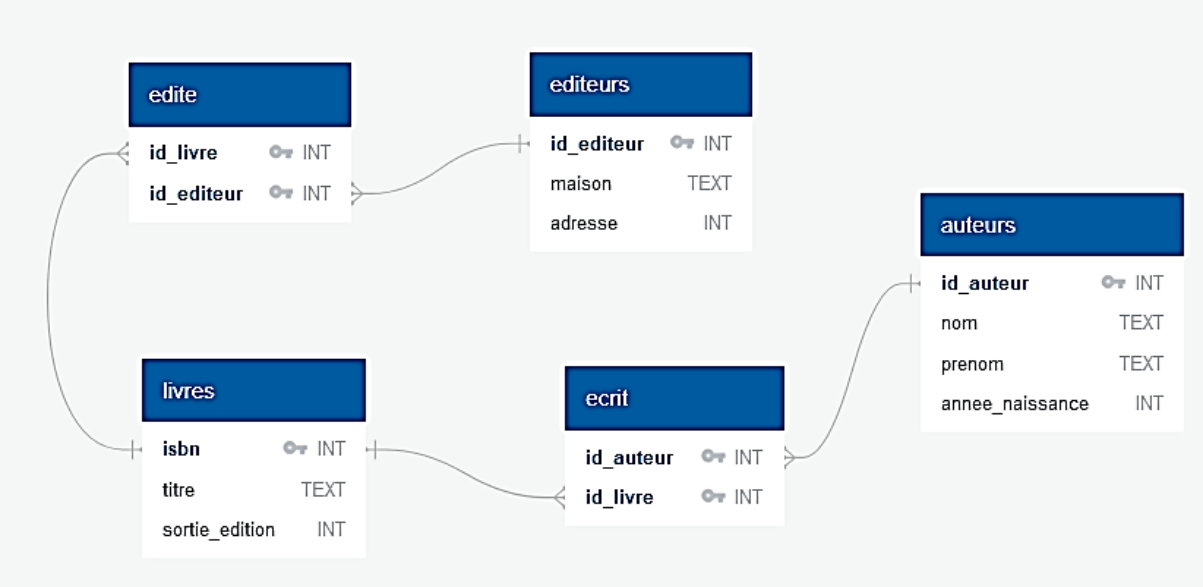
b. REQUETES SQL POUR INSERER CES NOUVEAUX ENREGISTREMENTS :

Requêtes 5. : Donner, dans le bon ordre (attention aux contraintes de *clé étrangère*), les 4 requêtes Sql, séparées par un ; , qui permettent d'insérer ces 2 nouveaux ouvrages. Exécuter ces requêtes sur DB Browser.

```
INSERT INTO auteurs(nom, prenom, annee_naissance) VALUES
('Villon','François',1463);

INSERT INTO livres VALUES
(13,'Le Lais',1972),
(14,'Le testament',1972);

INSERT INTO edite VALUES
(13,2),
(14,2);

INSERT INTO ecrit VALUES
(7,13),
(7,14);
```

3- CONTRAINTES DE CLE PRIMAIRE ET DE CLE ETRANGERE :

On rappelle ci-dessous le diagramme relationnel de la bdd.



Question 1 : Dans la relation (ou table) *edite*, la clé primaire est définie par un tuple regroupant les 2 attributs *id_livre* et *id_editeur*. Cette façon de faire permet de garantir que chaque enregistrement de cette table est unique. Est-il possible d'insérer plusieurs enregistrements pour un même livre, édité par plusieurs éditeurs ? … répondre dans le fichier Sql sous forme de commentaire selon le format donné ci-dessous :

```
--Question 1 : Un livre peut effectivement apparaitre dans plusieurs enregistrements
---------------de la table edite si l'editeur associé est différent. La contrainte de
---------------clé primaire impose uniquement de ne pas avoir plusieurs enregistrements
---------------avec un meme livre associé au meme editeur.
```

Question 2 : Dans la relation *ecrit*, la clé primaire est définie par un tuple regroupant les 2 attributs *id_livre* et *id_auteur*. Cette façon de faire permet de garantir que chaque enregistrement de cette table est unique. Est-il possible d'insérer plusieurs enregistrements pour un même livre, écrit par plusieurs auteurs ? … répondre dans le fichier Sql sous forme de commentaire.

4- QUELQUES REQUETES AVEC JOINTURE :

Requête 6. : Compléter ci-dessous la requête suivante qui permet de retourner un tableau contenant toutes les valeurs des attributs de la bdd, qui concernent Hugo :

```
--Requetes 6
SELECT * FROM editeurs
JOIN edite ON editeurs.          = edite.
JOIN livres ON edite.            = livres.
JOIN ecrit ON livres.            = ecrit.
JOIN auteurs ON auteurs.         = ecrit.
WHERE auteurs.       =
```

Résultat attendu dans DB Browser :

	id_editeur	maison	adresse	id_livre	id_editeur	isbn	titre	sortie_edition	id_auteur	id_livre	id_auteur	nom	prenom	annee_naissance
1	2	Livre de poche	43 rue de Grenelle, Paris 15	8	2	8	L'homme qui rit	2001	5	8	5	Hugo	Victor	1802
2	2	Livre de poche	43 rue de Grenelle, Paris 15	10	2	10	Hernani	1830	5	10	5	Hugo	Victor	1802
3	2	Livre de poche	43 rue de Grenelle, Paris 15	9	2	9	Les travailleurs de la mère	1842	5	9	5	Hugo	Victor	1802
4	7	Casterman	66 rue Bonaparte, Paris 6	9	7	9	Les travailleurs de la mère	1842	5	9	5	Hugo	Victor	1802

Donner ensuite les requêtes Sql qui permettent :

Requête 7. : de lister les ouvrages de Hugo :

```
SELECT titre FROM livres
JOIN ecrit ON livres.isbn = ecrit.id_livre
JOIN auteurs ON auteurs.id_auteur = ecrit.id_auteur
WHERE auteurs.nom = 'Hugo';
```

Résultat attendu dans DB Browser :

	titre
1	L'homme qui rit
2	Hernani
3	Les travailleurs de la mère

- ***Requête 8. :*** de lister les éditeurs de Hugo :

```
SELECT maison FROM editeurs
JOIN edite ON editeurs.id_editeur = edite.id_editeur
JOIN livres ON edite.id_livre = livres.isbn
JOIN ecrit ON livres.isbn = ecrit.id_livre
JOIN auteurs ON auteurs.id_auteur = ecrit.id_auteur
WHERE auteurs.nom = 'Hugo';
```

Résultat attendu dans DB Browser :

	maison
1	Livre de poche
2	Livre de poche
3	Livre de poche
4	Casterman

- ***Requête 9. :*** de lister les éditeurs de Hugo, mais sans mettre les doublons :

```
SELECT DISTINCT maison FROM editeurs
JOIN edite ON editeurs.id_editeur = edite.id_editeur
JOIN livres ON edite.id_livre = livres.isbn
JOIN ecrit ON livres.isbn = ecrit.id_livre
JOIN auteurs ON auteurs.id_auteur = ecrit.id_auteur
WHERE auteurs.nom = 'Hugo';
```

Résultat attendu dans DB Browser :

	maison
1	Livre de poche
2	Casterman

5- EXPLOITATION AVEC SQLITE3 ET PYTHON :

On se propose à présent d'exploiter cette base de données contenues dans le fichier *bibliotheque.db* avec le SGBD Sqlite3.

a. PHRASE 1 :

« Les ouvrages écrits par Hugo sont :
- L'homme qui rit édités par Livre de poche
- Hernani édités par Livre de poche
- Les travailleurs de la mère édités par Livre de poche
- Les travailleurs de la mère édités par Casterman »

On donne ci-après, le code python incomplet de la fonction *livreAuteur()* . L'exécution de cette fonction permet, après saisie d'un nom d'auteur, d'afficher la phrase précédente automatiquement. On donne en exemple une exécution réalisée avec l'auteur *Eddings* :

```
>>> (executing file "bibliotheque.py")
Entre un nom d'auteur : Eddings
Les ouvrages écrits par Eddings sont :
  'Le pion blanc des présages' édités par 'Presse Pocket'
  'La reine des sortilèges' édités par 'Presse Pocket'
  'Le réseau' édités par 'IRMA'
```

Ou avec un nom d'auteur qui n'est pas répertoriée dans cette bibliothèque :

```
>>> (executing file "bibliotheque.py")
Entre un nom d'auteur : Oscar
Aucun livre de Oscar dans cette bibliothèque
```

Lecture du fichier qui contient la bdd

```
import sqlite3
connexion = sqlite3.connect("bibliotheque.db")
curseur = connexion.cursor()
curseur.execute("PRAGMA foreign_keys = ON") # Active les clés étrangères

def livreAuteur() :
    nomSaisi = input("Entre un nom d'auteur : ")
    # Exécution des requêtes de sélection
    curseur.execute("""
                        SELECT livres.titre,editeurs.maison FROM editeurs
                        JOIN edite ON editeurs.id_editeur = edite.id_editeur
                        JOIN livres ON edite.id_livre = livres.isbn
                        JOIN ecrit ON livres.isbn = ecrit.id_livre
                        JOIN auteurs ON auteurs.id_auteur = ecrit.id_auteur
                        WHERE auteurs.nom = ? ;
                    """ , [nomSaisi])
    resultat = curseur.fetchall()
    print(resultat)
    # CODE A COMPLETER ICI

# Programme principal
livreAuteur()

connexion.close()
```

Le ou les ? qui sont dans la requête SQL prendront comme valeurs, dans l'ordre, celles des éléments de la liste placée ici.

⇨ Ouvrir Pyzo , ouvrir un nouveau fichier à enregistrer sous le nom *bibliotheque.py* et écrire ce script.

⇨ Analyser la structure de la liste renvoyée par l'exécution de la méthode *fetchall()* qui signifie en anglais : « ***récupérer tout*** ». Utiliser cette liste pour compléter le script de la fonction *livreAuteur()* afin de pouvoir retrouver l'affichage présenté en début de paragraphe.

CORRIGE

```
def livreAuteur() :
    nomSaisi = input("Entre un nom d'auteur : ")
    # Exécution des requêtes de sélection
    curseur.execute("""
                        SELECT livres.titre,editeurs.maison FROM editeurs
                        JOIN edite ON editeurs.id_editeur = edite.id_editeur
                        JOIN livres ON edite.id_livre = livres.isbn
                        JOIN ecrit ON livres.isbn = ecrit.id_livre
                        JOIN auteurs ON auteurs.id_auteur = ecrit.id_auteur
                        WHERE auteurs.nom = ? ;
                    """ , [nomSaisi])
    resultat = curseur.fetchall()
    #print(resultat)
    n = len(resultat)
    if n == 0 :
        print(f"Aucun livre de {nomSaisi} dans cette bibliothèque")
    else :
        print(f"Les ouvrages écrits par {nomSaisi} sont :")
        for e in resultat :
            print(f"  '{e[0]}' édités par '{e[1]}' ")
```

b. PHRASE 2 :

« Les ouvrages édités par Presse Pocket sont :

- 'Le pion blanc des présages'
- 'Métro pour l'enfer'
- 'La reine des sortilèges'»

Ecrire dans le fichier *bibliotheque.py*, le code python de la fonction *livreEditeur()* . L'exécution de cette fonction permet, après saisie d'un nom de maison d'édition, d'afficher la phrase précédente automatiquement. On donne 2 exemples d'exécution ci-dessous :

```
>>> livreEditeur()
Entre une maison d'édition : Presse Pocket
Les ouvrages édités par Presse Pocket sont :
  'Le pion blanc des présages'
  'Métro pour l'enfer'
  'La reine des sortilèges'

>>> livreEditeur()
Entre une maison d'édition : Branly
Aucun livre de Branly dans cette bibliothèque
```

CORRIGE

```
def livreEditeur() :
    maisonSaisie = input("Entre une maison d'édition : ")
    # Exécution des requêtes de sélection
    curseur.execute("""
                        SELECT livres.titre FROM editeurs
                        JOIN edite ON editeurs.id_editeur = edite.id_editeur
                        JOIN livres ON edite.id_livre = livres.isbn
                        JOIN ecrit ON livres.isbn = ecrit.id_livre
                        JOIN auteurs ON auteurs.id_auteur = ecrit.id_auteur
                        WHERE editeurs.maison = ? ;
                    """ , [maisonSaisie])
    resultat = curseur.fetchall()
    #print(resultat)
    n = len(resultat)
    if n == 0 :
        print(f"Aucun livre de {maisonSaisie} dans cette bibliothèque")
    else :
        print(f"Les ouvrages édités par {maisonSaisie} sont :")
        for e in resultat :
            print(f"  '{e[0]}'  ")
```

c. RECHERCHE D'UN TITRE AVEC SAISIE PARTIELLE DU TITRE :

Ecrire dans le fichier *bibliotheque.py*, le code python de la fonction *rechercheLivre()* . L'exécution de cette fonction permet, après saisie d'un nom de titre de livre, le nom pouvant être saisi partiellement, d'afficher la liste des ouvrages qui correspondent. On donne 2 exemples d'exécution ci-dessous :

```
>>> rechercheLivre()
Entre un nom de livre, même partiel : la
Les livres contenant la dans le titre sont
  'Le grand star blanc'
  'Lear/ la mer'
  'Le pion blanc des présages'
  'La reine des sortilèges'
  'Les travailleurs de la mère'

>>> rechercheLivre()
Entre un nom de livre, même partiel : zzz
Aucun livre avec zzz dans le titre
```

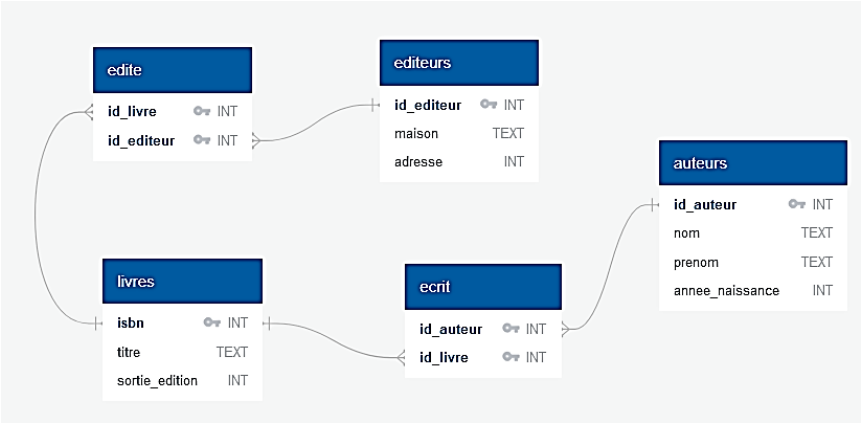
Le début du script python est le suivant :

```
def rechercheLivre() :
    saisie = input("Entre un nom de livre, même partiel : ")
    saisie2 = '%'+saisie+'%'
    # Exécution des requêtes de sélection
    curseur.execute("""
```

CORRIGE

```
def rechercheLivre() :
    saisie = input("Entre un nom de livre, même partiel : ")
    saisie2 = '%'+saisie+'%'
    # Exécution des requêtes de sélection
    curseur.execute("""
                        SELECT titre FROM livres
                        WHERE titre LIKE ?  ;
                    """ , [saisie2])
    resultat = curseur.fetchall()
    #print(resultat)
    n = len(resultat)
    if n == 0 :
        print(f"Aucun livre avec {saisie} dans le titre")
    else :
        print(f"Les livres contenant {saisie} dans le titre sont :")
        for e in resultat :
            print(f"  '{e[0]}'  ")
```

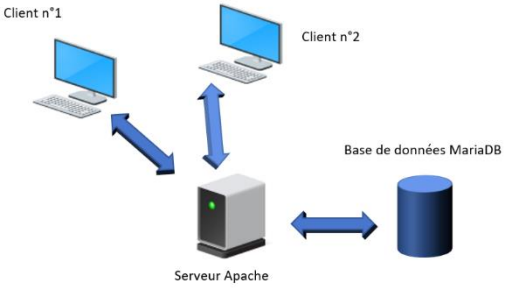
6- CONCLUSION ET FIN DU TP:



Cette activité vous a permis de pratiquer le langage Sql pour pouvoir exploiter les informations contenues dans une base de données constituées de 5 tables.

Les objectifs étaient également de découvrir une bdd moyennement complexe, avec des contraintes de clés primaires uniques pour 3 tables et sous forme de tuple de plusieurs attributs pour les 2 autres tables.

Cette bdd a été interrogée avec DB Browser dans un premier temps, puis avec le SGBD Sqlite3 utilisée dans un code python. On retrouve ainsi un fonctionnement proche de ce qui se passe sur un serveur qui héberge un site internet. Les requêtes sql sont construites en fonction des demandes qui émanent du client. La base de données hébergée sur le serveur retourne alors rapidement les informations demandées pour construire ensuite une réponse (fichier html ou code JS) qui est aussitôt envoyée au client.



⇨ N'oubliez pas d'uploader les fichiers *bibliotheque.py et bibliotheque.sql* sur nsibranly.fr .